Amendments to the Specification

The following refers to the paragraph numbering used in the application as published.

Please replace paragraph [0022] with the following replacement paragraph:

[0022] As may be seen in FIG. 2, the location of the conditional jump that is replaced is identified by code block a. The subroutine is identified as IRRITATE_1 (54) and includes code blocks identified as [[b, c, d and e]] b and c. The code block c includes a first and second sections 56 and 58, respectively. The start address of the second section 58 is predetermined and is indicated by the value KNOWN_DISPLACEMENT. The start address of the first section 56 is then determined by the difference between KNOWN_DISPLACEMENT and the upper limit of the distinguishing value V. The first section 56 consists of a series of conditional jumps to an address L1 and the second section 58 consists of a series of unconditional jumps to an address L2. The locations L1 and L2 contain code for returning program flow to execute statements1 and statements2 respectively. The code block b included in the subroutine IRRITATE_1 includes code for computing a difference between the KNOWN_DISPLACEMENT address and the THRESHOLD. The resulting address is then added to the distinguishing value V to derive a target address location in one of the sections 56 or 58.

Please replace paragraph [0033] with the following replacement paragraph:

[0033] It may be noted that the actual distinction between the two branches to be taken is decided at lines c18 and c22 where the retrieved subroutine return address is changed to the appropriate line in block a. In the present embodiment values of 0 and 1 have been chosen since the redirection jump instructions were located immediately after the call instruction to the subroutine IRRITATE_1, at lines a3 and a4 respectively. In other implementations different values with equal number of 1's in their binary presentation may be used so that [[an]] the difference in the add operations at lines c18 and c22 is indistinguishable to an attacker. In this case an appropriate number of NOP's would be added to code block a in order to adjust the return addresses.

Please replace paragraph [0036] with the following replacement paragraph:

[0036] As may be seen in FIG. 3, the location of the conditional jump that is replaced is identified by code block f. The subroutine is identified as IRRITATE_2 (102) and includes code blocks identified as blocks [[g and h]] g, h and i. The code block h also includes first and second sections 106 and 108, respectively. Each of the sections contain a series of dummy operations op1 indicated at lines h1 through h7 and at lines h12 through h18. Each of the sections is terminated by a sequence of instructions for retrieving the return address of the subroutine IRRITATE_2 and changing it such that the program counter will point to line f4 or f5 after returning from the subroutine. The lines f4 and f4 include jumps to one of the two branches indicates as block i and block j which contain statements1 and statements2 respectively.

Please replace paragraph [0037] with the following replacement paragraph:

[0037] The target destination address [[in]] is comprised of two components, namely the distinguishing value V or a value derived from V and a random number MASKED_RANDOM, that are added at line g1. The beginning address of the first and second sections are chosen such that this target address is either in the range of lines h1 through h8 or h12 through h19. Since, the second component of the target address is a random number, a random number of dummy operations will be executed before the return address of the subroutine IRRITATE_2 is computed at lines h8 to h10 (or h19 to h21).

Please replace paragraph [0042] with the following replacement paragraph:

[0042] For illustrative ~~purposed~~ purposes, we will in the following discussion assume [[an EC]] an elliptic curve (EC) scheme, where P is a point on the elliptic curve. The secret key d is normally combined with the point P to derive dP, the public key. However, the private key may also be used more frequently in various other cryptographic operations as described above. The cryptographic processor is generally initialized at manufacture time with the public key or secret value d. Initially, the value d may be divided into a number of parts, e.g. $d=b_{10}+b_{20}$.